(5) $\frac{a^n}{a^m}=a^{n-m}(a\neq 0)$；

(6) $a^{-m}=\frac{1}{a^m}(a\neq 0)$。

一个大于 1 的正整数，除了 1 和它自身外，不能被其他正整数整除的数叫作质数（又称为素数），否则就叫合数。1 既不是质数也不是合数。

质数的性质包括：

(1) 质数的个数是无穷的；

(2) 质数 p 的约数只有两个——1 和 p。

延伸阅读

[1] RONALD L G，DONALD E K，OREN P. 具体数学 计算机科学基础(第 2 版)[M]. 张明尧，张凡，译. 北京：人民邮电出版社，2013：85-103.

[2] THOMAS H C，CHARLES E L，RONALD L R，et al. 算法导论(原书第 3 版)[M]. 殷建平，徐云，王刚，译. 北京：机械工业出版社，2013：544-546.

典型题目

NOIP2012 普及组 质因数分解

（谷多玉　叶金毅）

1.5.3.2 取整

对于给定的实数 x，求与其接近的整数称为取整。常用的取整包括四舍五入取整、下取整、上取整和向零取整。

四舍五入取整用于取最接近实数 x 的整数，取整的结果为区间 $(x-0.5,x+0.5]$ 中的唯一整数。在数学中，一般用 $\approx$ 表示四舍五入取整。例如：$1.2\approx 1$，$-3.4\approx -3$，$4.5\approx 5$，$-4.5\approx -4$，$5.6\approx 6$，$-7.8\approx -8$。

在 C++中，`round`、`lround`、`llround` 等函数可实现四舍五入取整。

在 C++中，输出语句 `printf` 中可以使用`%.0lf` 对实数四舍五入取整后输出。例如 `printf("%.0lf",-7.8)`的输出为-8。

下取整又称向下取整，用于取不超过 x 的最大整数，取整结果为区间 $(x-1,x]$ 中的唯一整数。在数学中，一般用 $\lfloor x\rfloor$ 表示对 x 下取整。例如 $\lfloor 1.9\rfloor=1$，$\lfloor 2.1\rfloor=2$，$\lfloor -1.9\rfloor=-2$，$\lfloor -1.1\rfloor=-2$。

在 C++中，`floor` 函数可实现下取整。

上取整又称向上取整，用于取不小于 x 的最小整数，取整结果为区间 $[x,x+1)$ 中的唯一整数。在数学中，一般用 $\lceil x\rceil$ 表示对 x 上取整。例如 $\lceil 1.9\rceil=2$，$\lceil 2.1\rceil=3$，$\lceil -1.9\rceil=-1$，$\lceil -1.1\rceil=-1$。

在 C++中，`ceil` 函数可实现上取整。

向零取整又称去尾取整，用于取 0 和 x(含)之间最接近 x 的整数。当 $x\geqslant 0$ 时，对 x 向零

取整的结果与下取整的结果相同；当 $x<0$ 时，对 x 向零取整的结果与上取整的结果相同。

在 C++中，强制类型转换可实现向零取整，例如 `(int)2.9` 的值为 2。

在 C++中，整除运算的结果等于将商向零取整的结果。

（胡伟栋）

1.5.3.3 模运算与同余

模运算：对于给定的整数 a 和整数 b，a 对 b 的模为 0(含)和 b(不含)之间的整数 m，使得 $a-m$ 为 b 的整数倍。a 对 b 的模记为 $a \bmod b$，符号与 b 的符号相同。

模运算与求余运算在 a 和 b 为正整数时结果完全相同，因此在信息学中一般不严格区分模运算与求余运算。当 a 或 b 中出现负数时模运算和求余的结果可能不同。

同余：给定一个正整数 m，如果两个整数 a 和 b 除以 m 的余数相同，即 $a \bmod m = b \bmod m$，则称 a 和 b 对于 m 同余，m 称为同余的模。同余的概念也可以这样理解：$a-b$ 是 m 的整倍数，也就是 $m \mid (a-b)$。同余记作 $a \equiv b \pmod m$。同余的性质如下。

(1) 自反性：$a \equiv a \pmod m$。

(2) 对称性：如果 $a \equiv b \pmod m$，则 $b \equiv a \pmod m$。

(3) 传递性：如果 $a \equiv b \pmod m$，$b \equiv c \pmod m$，则 $a \equiv c \pmod m$。

(4) 可加性：如果 $a \equiv b \pmod m$，$c \equiv d \pmod m$，则 $a \pm c \equiv b \pm d \pmod m$。

(5) 可乘性：如果 $a \equiv b \pmod m$，$c \equiv d \pmod m$，则 $ac \equiv bd \pmod m$。

(6) 对于任意自然数 n，如果 $a \equiv b \pmod m$，则 $an \equiv bn \pmod m$。

(7) 如果 $ac \equiv bc \pmod m$，$(c,m)=1$，那么 $a \equiv b \pmod m$，其中 $(c,m)=1$ 表示 c 与 m 的最大公约数为 1。

参考词条

算术运算：加、减、乘、除、整除、求余

延伸阅读

[1] RONALD L G，DONALD E K，OREN P. 具体数学 计算机科学基础(第 2 版)[M]. 张明尧，张凡，译. 北京：人民邮电出版社，2013：85-103.

[2] THOMAS H C，CHARLES E L，RONALD L R，et al. 算法导论(原书第 3 版)[M]. 殷建平，徐云，王刚，译. 北京：机械工业出版社，2013：544-546.

（谷多玉　叶金毅）

1.5.3.4 整数唯一分解定理

唯一分解定理又称作算术基本定理，对于任何一个大于 1 的自然数 n，要么 n 本身是质数，要么可以分解为 2 个或者 2 个以上的质数的乘积，而且分解方法唯一。n 的标准分解式可以写为：

$$n = p_1^{a_1} p_2^{a_2} \cdots p_m^{a_m}$$

其中，p_i 为质因子，a_i 为指数。

代码示例

整数唯一分解定理的核心代码如下。

```
int p[105],a[105],tot = 0; // p 数组存储质因子,a 数组存储对应质因子的指数,tot 表示
                              质因子的个数
void getfac(int x)
{
    for (int i = 2; i * i <= x; i++)
    {
        if (x %i == 0)
        {
            p[++tot] = i;
            while (x %i == 0)
            {
                a[tot]++;
                x /= i;
            }
        }
    }
    if (x > 1)
    {
        p[++tot] = x;
        a[tot] = 1;
    }
}
```

参考词条

整除、因数、倍数、指数、质(素)数、合数

延伸阅读

[1] RONALD L G, DONALD E K, OREN P. 具体数学 计算机科学基础(第 2 版)[M]. 张明尧, 张凡, 译. 北京: 人民邮电出版社, 2013: 88-89.

[2] KENNETH H R. 初等数论及其应用(原书第 6 版)[M]. 夏鸿刚, 译. 北京: 机械工业出版社, 2015: 82-92.

[3] THOMAS H C, CHARLES E L, RONALD L R, et al. 算法导论(原书第 3 版)[M]. 殷建平, 徐云, 王刚, 译. 北京: 机械工业出版社, 2013: 546.

典型题目

1. NOIP2012 普及组 质因数分解
2. NOIP2009 提高组 Hankson 的趣味题

(谷多玉　叶金毅)

1.5.3.5 辗转相除法

辗转相除算法又称为欧几里得算法，用于求两个非负整数的最大公约数(Greatest Common Divisor，GCD)。

给定非负整数 a 和 b，它们最大公约数的计算公式为 $\gcd(a,b)=\gcd(b,a\%b)$，其中 $\gcd(a,b)$ 表示 a 和 b 的最大公约数。

代码示例

辗转相除算法的核心代码如下。

```
int gcd(int a,int b)
{
    if (b == 0)
        return a; //边界条件,如果 b 为 0,则最大公因数为 a
    return gcd(b,a %b); //辗转相除
}
```

参考词条

1. 整除、因数、倍数、指数、质(素)数、合数
2. 扩展欧几里得算法

延伸阅读

[1] KENNETH H R. 初等数论及其应用(原书第 6 版)[M]. 夏鸿刚，译. 北京：机械工业出版社，2015：74-80.

[2] THOMAS H C，CHARLES E L，RONALD L R，et al. 算法导论(原书第 3 版)[M]. 殷建平，徐云，王刚，译. 北京：机械工业出版社，2013：547-549.

典型题目

1. NOIP2001 普及组 最大公约数和最小公倍数问题
2. NOIP2009 提高组 Hankson 的趣味题

(谷多玉　叶金毅)

1.5.3.6 素数筛法：埃氏筛法与线性筛法

埃氏筛法(the sieve of Eratosthenes，埃拉托色尼筛)是一种古老而简单的方法，可以找到 $[2,n]$ 范围内的所有素数。埃氏筛法的核心思想是一个素数的倍数一定是合数。对于初始序列 2,3,4,5,6,7,8,9,10,11,12,13,⋯，操作步骤如下。

(1) 首先筛去素数 2 的倍数。

$$2,3,\cancel{4},5,\cancel{6},7,\cancel{8},9,\cancel{10},11,\cancel{12},13,\cdots$$

(2) 在之后未筛除的数中选择第一个数，即素数 3，再筛去 3 的倍数。

$$2,3,\cancel{4},5,\cancel{6},7,\cancel{8},\cancel{9},\cancel{10},11,\cancel{12},13,\cdots$$

（3）在之后未筛除的数中选择第一个数，即素数 5，再筛去 5 的倍数。

$$2,3,\cancel{4},5,\cancel{6},7,\cancel{8},\cancel{9},\cancel{10},11,\cancel{12},13,\cdots$$

依次类推，直到 n 中所有素数的倍数都被筛除，留下的未被筛选的就是 n 之内的素数。本算法的时间复杂度为 $O(n\log\log n)$。

线性筛法，又称为欧拉筛法，其核心思想是，每个合数只被最小的质因子筛掉，或者说被最大的因子筛掉。埃氏筛法中有些合数被重复筛除，例如 $12=2^2\cdot 3$，会被 2 筛除一次，又会被 3 筛除一次。欧氏筛法中每个合数只被最小的质因子或者说被最大的真因子筛除，确保每个数只被筛一次，实现过程为如下。

对于每个数 i，将 i 的质数倍筛除，设置 $i \bmod \mathrm{prime}[j]=0$ 时停止，不再往后筛除 i 的质数倍。

线性筛法的原理是，因为 $i \bmod \mathrm{prime}[j]=0$，则可设 $i=\mathrm{prime}[j]\cdot a$（其中 a 为正整数），则对于第 $j+1$ 个素数 $\mathrm{prime}[j+1]$，$i\cdot\mathrm{prime}[j+1]$ 可写为 $i\cdot\mathrm{prime}[j+1]=\mathrm{prime}[j]\cdot a\cdot\mathrm{prime}[j+1]$，显然 $a\cdot\mathrm{prime}[j+1]$ 是比 $a\cdot\mathrm{prime}[j]$ 更大的数，则 $i\cdot\mathrm{prime}[j+1]$ 可以被一个比 i 更大的因子乘以质数倍筛除。所以如果 $i \bmod \mathrm{prime}[j]=0$，则以 i 为最大约数的筛数结束。

例如已经筛到 7，现有质数分别是 2，3，5，7，数字 8 可以筛掉 $8\times2=16$，因为 $8 \bmod 2=0$，所以使用数字 8 的筛数结束，如果继续向下筛，则 8 不会是继续筛掉数值的最大因子；数字 9 可以筛掉 $9\times2=18$，$9\times3=27$，因为 $9 \bmod 3=0$，所以使用数字 9 的筛数结束；后续数字以此类推。本算法的时间复杂度为 $O(n)$。

代码示例

埃氏筛法的核心代码如下。

```
const int MAXN = 1e5 + 5;
int prime[MAXN],cnt;
bool flag[MAXN];
void E_Sieve(int n)
{
    for (int i = 2; i <= n; i++)
    {
        if (!flag[i])
        {
            for (int j = 2; j * i <= n; j++) //筛除素数 i 的所有倍数
                flag[j * i] = 1;
        }
    }
    for (int i = 2; i <= n; i++)
        if (!flag[i])
            prime[++cnt] = i;
}
```